PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F

A2

(11) International Publication Number: WO 99/60459

(43) International Publication Date: 25 November 1999 (25.11.99)

(21) International Application Number: PCT/US99/10942

(22) International Filing Date: 18 May 1999 (18.05.99)

(30) Priority Data:
09/081,860 19 May 1998 (19.05.98) US

(71) Applicant: SUN MICROSYSTEMS, INC. [US/US]; 901 San Antonio Road, M/S: UPAL01-521, Palo Alto, CA 94303 (US).

(72) Inventors: GUPTA, Amit; 34077 Paseo Padre Parkway #106, Fremont, CA 94555 (US). BAEHR, Geoffrey, A.; 531 Colorado Avenue, Palo Alto, CA 94306 (US). ROM, Raphael; 69 Roosevelt Circle, Palo Alto, CA 94306 (US). SCHUBA, Christoph; 473 Hope Stree #1, Mountain View, CA 94041 (US).

(74) Agents: HECKER, Gary, A. et al.; Hecker & Harriman, Suite 2300, 1925 Century Park East, Los Angeles, CA 90067 (US).

(81) Designated States: AE, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
Without international search report and to be republished upon receipt of that report.

(54) Title: METHOD AND APPARATUS FOR EFFECTIVE TRAFFIC LOCALIZATION THROUGH DOMAIN NAME SYSTEM

(57) Abstract

The present invention uses a client-side computation to efficiently provide translation of a domain name to the address of a "good" (i.e. close, available, nearby) server of a distributed server system. The invention uses an application client's resolver to perform some computation to determine the IP address of a preferred server for that client. When the client provides the web server name (say www.sun.com) to the DNS resolver, the DNS service returs data, or a small applet that runs at the browser's local resolver to generate the desired IP address. The present invention is processed by the DNS server (in the resolver portion) and at the client. The web server does not need to participate. The invention does no require any changes to the current DNS infrastructure. The invention also can direct clients to more local servers and avoid expensive "long-haul" links. The invention also preserves the critical caching property of the current DNS system, has reduced latency than other schemes, less traffic for the network and DNS servers, and supports the use of secondary DNS servers.

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	ES	Spain	LS	Lesotho	SI	Slovenia
AM	Armenia	FI	Finland	LT	Lithuania	SK	Slovakia
AT	Austria	FR	France	LU	Luxembourg	SN	Senegal
AU	Australia	GA	Gabon	LV	Latvia	SZ	Swaziland
AZ	Azerbaijan	GB	United Kingdom	MC	Monaco	TD	Chad
BA	Bosnia and Herzegovina	GE	Georgia	MD	Republic of Moldova	TG	Togo
BB	Barbados	GH	Ghana	MG	Madagascar	TJ	Tajikistan
BE	Belgium	GN	Guinea	MK	The former Yugoslav Republic of Macedonia	TM	Turkmenistan
BF	Burkina Faso	GR	Greece			TR	Turkey
BG	Bulgaria	HU	Hungary	ML	Mali	TT	Trinidad and Tobago
BJ	Benin	IE	Ireland	MN	Mongolia	UA	Ukraine
BR	Brazil	IL	Israel	MR	Mauritania	UG	Uganda
BY	Belarus	IS	Iceland	MW	Malawi	US	United States of America
CA	Canada	IT	Italy	MX	Mexico	UZ	Uzbekistan
CF	Central African Republic	JP	Japan	NE	Niger	VN	Viet Nam
CG	Congo	KE	Kenya	NL	Netherlands	YU	Yugoslavia
CH	Switzerland	KG	Kyrgyzstan	NO	Norway	ZW	Zimbabwe
CI	Côte d'Ivoire	KP	Democratic People's Republic of Korea	NZ	New Zealand		
CM	Cameroon			PL	Poland		
CN	China	KR	Republic of Korea	PT	Portugal		
CU	Cuba	KZ	Kazakstan	RO	Romania		
CZ	Czech Republic	LC	Saint Lucia	RU	Russian Federation		
DE	Germany	LI	Liechtenstein	SD	Sudan		
DK	Denmark	LK	Sri Lanka	SE	Sweden		
EE	Estonia	LR	Liberia	SG	Singapore		

METHOD AND APPARATUS FOR EFFECTIVE TRAFFIC LOCALIZATION THROUGH DOMAIN NAME SYSTEM

BACKGROUND OF THE INVENTION

1. FIELD OF THE INVENTION

This invention relates to the field of managing communication on the internet.

2. BACKGROUND ART

Computer systems sometimes rely on a server computer system to provide information to requesting computers on a network. When there are a large number of requesting computers, it may be necessary to have more than one server computer system to handle the requests. In prior art systems, there is a problem in efficiently directing requests to the correct server in a multiple server system.

One area where this has been a problem is on the internet. The problem can be better understood by reviewing the structure and operation of the internet below.

The Internet

The Internet is a worldwide network of interconnected computers. An Internet client accesses a computer on the network via an Internet provider. An Internet provider is an organization that provides a client (e.g., an individual or other organization) with access to the Internet (via analog telephone line or Integrated Services Digital Network line, for example). A client can, for example, read information from, download a file from or send an electronic mail message to another computer/client using the Internet.

To retrieve a file or service on the Internet, a client must search for the file or service, make a connection to the computer on which the file or service is stored, and download the file or service. Each of these steps may involve a separate application and access to multiple, dissimilar computer systems. The World Wide Web (WWW) was developed to provide a simpler, more uniform means for accessing information on the Internet.

The components of the WWW include browser software, network links, servers. and WWW protocols. The browser software, or browser, is a user-friendly interface (i.e., front-end) that simplifies access to the Internet. A browser allows a client to communicate a request without having to learn a complicated command syntax, for example. A browser typically provides a graphical user interface (GUI) for displaying information and receiving input. Examples of browsers currently available include Mosaic, Netscape Navigator and Communicator, Microsoft Internet Explorer, and Cello.

Information servers maintain the information on the WWW and are capable of processing a client request. Hypertext Transport Protocol (HTTP) is the standard protocol for communication with an information server on the WWW. HTTP has communication methods that allow clients to request data from a server and send information to the server.

To submit a request, the client contacts the HTTP server and transmits the request to the HTTP server. The request contains the communication method requested for the transaction (e.g., GET an object from the server or POST data to an object on the server). The HTTP server responds to the client by sending a status of the request and the requested information. The connection is then terminated between the client and the HTTP server.

A client request therefore, consists of establishing a connection between the client and the HTTP server, performing the request, and terminating the connection. The HTTP server does not retain any information about the request after the connection has been terminated. HTTP is, therefore, a stateless protocol. That is, a client can make several requests of an HTTP server, but each individual request is treated independent of any other request. The server has no recollection of any previous request.

An addressing scheme is employed to identify Internet resources (e.g., HTTP server, file or program). This addressing scheme is called Uniform Resource Locator (URL). A URL contains the protocol to use when accessing the server (e.g., HTTP), the Internet domain name of the site on which the server is running, the port number of the server, and the location of the resource in the file structure of the server.

The WWW uses a concept known as hypertext. Hypertext provides the ability to create links within a document to move directly to other information. To activate the link, it is only necessary to click on the hypertext link (e.g., a word or phrase). The hypertext link can be to information stored on a different site than the one that supplied the current information. A URL is associated with the link to identify the location of the additional information. When the link is activated, the client's browser uses the link to access the data at the site specified in the URL.

If the client request is for a file, the HTTP server locates the file and sends it to the client. An HTTP server also has the ability to delegate work to gateway programs. The Common Gateway Interface (CGI) specification defines a mechanism by which HTTP servers communicate with gateway programs. A gateway program is referenced using a URL. The HTTP server activates the program specified in the URL and uses CGI mechanisms to pass program data sent by the client to the gateway program. Data is passed from the server to the gateway program via command-line arguments, standard input, or environment variables. The gateway program processes the data and returns its response to the server using CGI (via standard input, for example). The server forwards the data to the client using the HTTP.

A browser displays information to a client/user as pages or documents (referred to as "web pages" or "web sites"). A language is used to define the format for a page to be displayed in the WWW. The language is called Hypertext Markup Language (HTML). A WWW page is transmitted to a client as an HTML document. The browser executing at the client parses the document and displays a page based on the information in the HTML document.

HTML is a structural language that is comprised of HTML elements that are nested within each other. An HTML document is a text file in which certain strings of characters, called tags, mark regions of the document and assign special meaning to them. These regions are called HTML elements. Each element has a name, or tag. An element can have attributes that specify properties of the element. Blocks or components include unordered list, text boxes, check boxes, radio buttons, for example. Each block has properties such as name, type, and value. The following provides an example of the structure of an HTML document:

```
<HTML>
    <HEAD>
    .... element(s) valid in the document head
    </HEAD>
    <BODY>
    .... element(s) valid in the document body
    </BODY>
</HTML>
```

Each HTML element is delimited by the pair of characters "<" and ">". The name of the HTML element is contained within the delimiting characters. The combination of the name and delimiting characters is referred to as a marker, or tag. Each element is identified by its marker. In most cases, each element has a start and ending marker. The ending marker is identified by the inclusion of an another character, "/" that follows the "<" character.

HTML is a hierarchical language. With the exception of the HTML element, all other elements are contained within another element. The HTML element encompasses the entire document. It identifies the enclosed text as an HTML document. The HEAD element is contained within the HTML element and includes information about the HTML document. The BODY element is contained within the HTML. The BODY element contains

all of the text and other information to be displayed. Other HTML elements are described in HTML reference manuals.

Domain Name Server

A computer user navigates the internet or web from a browser on a computer system. To access a web site, the user enters the host name of the web site into the browser. This can be accomplished by clicking on a link, by activating a tool bar button, or by manually entering a name or address into a location field and pressing "enter". The names that a browser client uses are known as host names, such as www.sun.com for example. The name that is entered is not the actual Internet Protocol (IP) address of the intended web server. The actual IP address is a string of numbers that uniquely locate the web server that provides the web site data. A worldwide distributed database system, called the "Domain Name System (DNS)" provides the mapping between server names and the associated IP addresses.

Client application software, such as a web browser, use a local library, called the "DNS resolver" to obtain the translation from server name to IP address. The resolver in turn contacts a predetermined local DNS server to obtain the translation. DNS servers can maintain caches of previously resolved names. More specifically, name resolution processes typically require two hosts on the client side. Consider a user working on "asha.eng.sun.com" that wants to get then address of "whitehouse.com". The client browser will talk with a local resolver (a library attached to the browser process itself, in the current example running on asha.eng.sun.com). The local resolver will go to one of a relatively small number of local name servers, e.g. "ns.sun.com". Here ns.sun.com is called the client side name server. The client side name server will communicate with the outside

world to determine the IP address of whitehouse.com, and forward this information to the resolver that is part of the browser process.

DNS is a global network of servers that translate host names into numerical addresses (known as Internet Protocol, or IP addresses) and provides IP address to name mapping as well. A DNS server consists of a name server and a resolver. The name server provides responses to resolver requests when it can by supplying the correct address for the host name supplied by the resolver. When the DNS server is unable to provide the correct address, it invokes its resolver to provide a solution. The resolver passes the unknown host name to another name server on the internet network and waits for a reply. If the next name server can provide the address for the host name, it does. Otherwise, the host name is forwarded to another name server, and this repeats until a name server is found that can translate the host name, or until it is determined that no translation exists. The server with the correct translation (if any) then forwards it to the server that requested it, and that server returns it to the server that requested it from the second server, and so on until it is returned to the original name server, who can then return it to the client's resolver, who returns it to the browser. The originally contacted name server then stores the translation in a local cache so that the next time it sees the host name, it can supply a translation without asking another name server.

Once the IP address is known, the browser communicates with the web server at that address to retrieve the requested web page or other information.

The operation of the DNS network is described in:

P.V. Mockapetris "Domain names - concepts and facilities", RFC 1034. Nov 1987.

P.V. Mockapetris "Domain names - implementation and specification", RFC 1035. Nov 1987.

DNS Server Problems

An internet server is typically limited IN the number of clients it can efficiently service at any one time. However, the owners of an internet site do not want users to be denied access to their internet site. If the site is popular, internet site owners desire all attempted accesses to be successful.

To provide such service, some companies have implemented systems that allow multiple internet servers to service requests for a single internet site. If there are two servers, it would be expected that each server would service approximately half of the client requests to the supported internet site. This concept is known as "distributed servers", in our case, "distributed internet servers".

There are a number of schemes for implementing the distributed internet server. The schemes involve the manner in which requests to the internet site address are routed to the multiple server. One such scheme is called a DNS shuffle address or "round-robin". In this scheme, as each request comes to the internet site address, the servers that respond are rotated in some order. If there are three servers in the distributed system, then any one of the servers handles every third request. This scheme has a disadvantage of ignoring load balancing considerations and traffic localization considerations.

Another scheme uses a freely available script called "lbnamed" that provides a DNS server with the ability to return a different IP address for every client request received for a internet site host address. The returned IP

address can be made to depend on server load as well as availability of local internet servers, but ignores the relative distance between clients and the available servers.

A product by Cisco Systems known as "DistributedDirector" is a scheme that relies on internet routing tables to provide locality information. The Cisco scheme imposes extra latency during servicing of DNS requests. It also suppresses DNS caching mechanisms, adding traffic to the internet. It also reacts to each change in the internet routing, which changes very frequently.

IBM has developed an "Interactive Network Dispatcher (IND) load balancing product. It consists of Interactive Session Support (ISS) and a Network Dispatcher (ND). For TCP/IP client requests, IND chooses a server cluster (via ISS) and then directs the client request to the appropriate server (via ND). ND routes the request to the chosen server transparently. The ISS can generate load information on servers, can perform ping triangulation initiated at servers to determine the "nearest" server (cluster) to a client, and influence client routing of requests by supplying the necessary DNS replies. Load information is collected through load monitoring agents (advisors) near the servers. Multiple metrics are supported (e.g., CPU, DASD, I/O). ISS provides its own DNS server implementation to generate the necessary DNS replies. This scheme has the disadvantage of requiring modification of the existing DNS system. There are also disadvantages associated with ping triangulation. These include increased network traffic, timing penalties in performing the ping operation, and the fact that it needs to be updated continuously to maintain accuracy.

Another approach is known as "smart clients" from U. C. Berkeley Research. The smart client approach is an architecture for web traffic client-server communication that allows for a dynamic server choice based on load and availability. The approach allows for the use of multiple server machines to achieve scalable performance, for load balancing, for fault transparency, and backwards compatibility with the existing addressing scheme (URLs). The architecture requires client web browsers to execute downloadable, service specific code. This code is divided into a GUI thread and director thread. Server choice, load balancing, and fault transparency are encapsulated by the director thread. A disadvantage of this scheme is that it requires cooperation of requesting clients. It imposes extensive overhead on single web page retrieval and surfing to new sites.

Prior art scheme such as the ones described above are described in the following:

"A Novel Server Selection Technique for Improving the Response Time of a Replicated Service" Zongming Fei, Samrat Bhattacharjee, Ellen W. Zegura, Mostafa H. Ammar. Networking and Telecommunications Group, College of Computing, Georgia Institute of Technology.

"Chapter 1. Introducing IBM's Interactive Network Dispatcher" IBM Users Guide. (http://www.software.ibm.com/enetwork/dispatcher/)

"Cisco DistributedDirector • More Information on DistributedDirector" (http://www.cisco.com/warp/public/751/distdir/dd_wp.htm)

"Using Smart Clients to Build Scalable Services" Chad Yoshikawa, Brent Chun, Paul Eastham, Amin Vahdat, Thomas Anderson, and David Culler. Computer Science Division, University of California, Berkeley

SUMMARY OF THE INVENTION

The present invention uses a client-side computation to efficiently provide translation of a domain name to the address of a "good" (i.e. close, available, nearby) server of a distributed server system. The invention uses an application client's resolver to perform some computation to determine the IP address of a preferred server for that client. When the client provides the web server name (say www.sun.com) to the DNS resolver, the DNS service returns data, or a small applet that runs at the browser's local resolver to generate the desired IP address. The present invention is processed by the DNS server (in the resolver portion) and at the client. The web server does not need to participate. The invention does not require any changes to the current DNS infrastructure. The invention also can direct clients to more local servers and avoid expensive "long-haul" links. The invention also preserves the critical caching property of the current DNS system, has reduced latency than other schemes, less traffic for the network and DNS servers, and supports the use of secondary DNS servers.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a block diagram of an example computer system that can be used with the present invention.

Figure 2 is a diagram illustrating an embodiment of the present invention.

Figure 3A is a diagram illustrating the operation of an embodiment of the present invention.

Figure 3B is a flow diagram illustrating the operation of an embodiment of the present invention.

DETAILED DESCRIPTION OF THE INVENTION

The invention is a method and apparatus for providing effective traffic localization through the Domain Name System. In the following description, numerous specific details are set forth to provide a more thorough description of embodiments of the invention. It will be apparent, however, to one skilled in the art, that the invention may be practiced without these specific details. In other instances, well known features have not been described in detail so as not to obscure the invention.

The invention provides a system for providing IP addresses of servers in a distributed server system in a manner that provides efficient traffic localization. The invention takes advantage of the existing DNS infrastructure and uses some client processing for server selection. When a browser client provides an address to a DNS server, the server responds, when appropriate, with data, a table of data, or a thin client applet to the client browser. The client's local resolver uses the data, table or applet to generate the IP address of a server that is a "best" server for the client. The selected server may not be the optimal server for the client, but it may be one that promotes efficient use of network resources. The invention has equal application when the DNS server provides either data, a table of data, or an applet. For purposes of discussion, the information will generally be referred to here as "table/applet". Where appropriate, specific discussion of one of the possible forms of the information will be made. While the ensuing discussion uses the web service as a canonical example of an Internet service, the invention described herein applies equally well to all other internet services, including, but not limited to, the telnet service and the ftp service.

The invention is implemented via agents, servers, and clients. The agents collect the network topology/load information at a particular web server farm, and provide it to the DNS server for this service. (Here, a web server farm refers to a set of nearby servers that work together). The servers obtain the load/topology information from many agents, and use the information to generate a single applet/table. The clients employ the applet/table to choose from the available IP addresses for the given server name.

A diagram of the present invention is illustrated in Figure 2. In the example of Figure 2, a distributed server system serves the web site "www.sun.com". Referring to Figure 2, two web servers 201A and 201B both server the web site www.sun.com. These web servers feed load/topology information about themselves to new tables/applets database 202. This database is coupled to the Sun primary DNS server 205. The Sun Primary DNS server 205 includes a resolver library 206 and is coupled to a DNS database 203 and cache 204. The Sun primary DNS server 205 handles requests from other resolvers such as request 207. This request can come from a client browser resolver, or from other DNS servers. The DNS server 205 handles the request and sends reply 208.

When DNS server 205 receives a request to resolve the host name sun.com into an IP address, it provides a table/applet as a reply 208 to the requesting resolver.

The distance/metric tables are created by agents collecting network topology and load information. This is done at a particular web server farm and the information is provided to the DNS server that serves the web site.

The agent is a process that is used by any distributed server system that desires to take advantage of the present invention.

The agents could use any of a number of processes to obtain the necessary information. For example, IBM uses ping triangulation to obtain approximate delay information. One description of collecting information from ping requests and incorporated herein by reference is described at http://www.internetweather.com based on 45 packet (one packet/second) ping tests with 210 byte packets between a central site and the primary domain server for a number of networks. Cisco uses BGP information from its own routers to get this information. Another way is to get the picture from routing tables at major exchange points (for example, the web site www.merit.edu provides routing table snapshots). Links to information regarding these methods may be found at the Cooperative Association for Internet Data Analysis (CAIDA) at http://www.caida.org and incorporated herein by reference.

The information does not have to be precise. It may be possible to define ranges of IP addresses that define groups of servers. Consider where servers are defined in groups A, B, and C. Information that shows that server cluster hosts in group A are served by same ISP, hosts in group B have to go through one major exchange, and those in group C have to go through expensive, under-provisioned international routes are usable in the present invention.

For a load value, the machine load average may be used. Alternatively, it may be suitable to measure the weighted tail average of connection set up rate, or the disk I/O rate, and normalize this by the machine capacity. Another option would be to periodically connect to the

server process from a nearby client (or same machine), and use the time-to-connect as an estimate of the machine load. This can also be used to calibrate the server when using other metrics (connection set-up average, disk I/O etc.)

Servers (Collating Topology/Load Information)

The servers are responsible for collating the topology/load information that the agents provide and to use this information to generate the table/applet. By way of example, let us assume that there are two servers in a distributed server system, with IP addresses 11.11.11.11 and 22.22.22.22 respectively. The agents could provide a table for each server as shown in Tables 1 and 2 below.

Table 1 (Server 11.11.11.11)

Client	Distance metric
net x.y.0.0	4
net x.z.0.0	3
net a.b.c.0	6

Table 2 (Server 22.22.22.22)

Client	Distance metric
net x.y.0.0	2
net x.z.0.0	5
net a.b.c.0	3

The server could merge together this information by going through all the client tables in parallel and working through the different address blocks. For each block, the server could choose the best server of the distributed servers through a combination of locality and load information. This information can be placed in a generated table which is returned in a new

DNS record type. One example of such a merged table is illustrated below in Table 3.

Table 3

Client	Server
net x.y.0.0	Srv 22.22.22.22
net x.z.0.0	Srv 11.11.11.11
net a.b.c.0	Srv 22.22.22.22

One option would be to restrict the generated table to provide only one choice to each client, depending on the client IP address. For example, if the client IP address is within a certain range, one server in the server farm would be identified as the appropriate server for the web site. Another option is to allow clients a small number of choices. The client can choose nearness or locality in "hard" cases. Letting a client choose among several choices may provide a degree of randomization that leads to improved balancing among several nearby servers. An alternate and more dynamic option would have the applet periodically query a "load-server" to refresh the table information. (Note that in these cases, the server should manage the TTL information that it returns to the resolver).

It is useful for servers to restrict the size of the tables that they return to the clients. For backward compatibility with existing resolvers, the server should also put one (or more) IP addresses in the DNS address record returned.

Clients (Server Choice)

The client-side code resides in the resolver. It is responsible for sending the DNS lookup request to the DNS server, accepting the returned applet/table from the DNS server, to run the applet/table code to choose the

IP address that the server name maps to, and to return this IP address to the requesting application. This component should be the simplest part of this system - all it depends on is the language used for shipping the condensed server load/locality information in the applet/table. If the applet is a Java™ applet, this component just executes the Java™ programming language bytecode. If the returned information is a simple table, this component looks up the table to find the entry that corresponds to the client IP address.

In an embodiment of the invention, the information distributed by the DNS is limited to the full set of server IP addresses. Clients make the choice of which server to contact after the DNS resolution step, before the actual request is made. There are at several options to how clients can make this choice, including, but not limited to, client initiated ping triangulation, client initiated timing service request, random choice, cost-based choice, or no choice (where the system provides only one address, or the client always chooses just one).

The client initiated ping triangulation may suffer from measurement problems.

The client initiated timing approach is to send a minimal service request to all servers listed in the table and measure the response time. For example, in absence of a standard timing service request, a Web client could request the retrieval of a file that would not be served (e.g.,/dev/null) and measure the arrival of the corresponding HTTP error message. This approach suffers from some of the ping triangulation problems, but has no firewall filtering problems, for example. This approach gains in attractiveness if the availability of HTTP over hybrid TCP-UDP is assumed. Furthermore, in the scenario where the IP addresses represent server clusters,

the fake request would be intercepted by the network dispatcher and its reply could be artificially delayed based on the load characteristics present at the network dispatcher.

Figure 3A illustrates the relationship between the client browser and the DNS server in an embodiment of the present invention. The client browser 390 has an associated DNS resolver 391. When a host name is entered in the browser 390, the DNS resolver 391 initiates a request to a DNS server 392 for translation. The DNS server 392 replies to the DNS resolver 391 with an IP address or with a table/applet.

The operation of the components of Figure 3A is illustrated in the flow diagram of Figure 3B. At step 301 a browser user or application enters a host name into the browser. At step 302, the browser requests a translation of the host name from its DNS resolver. At decision block 303 the argument "Name Cached?" is made. This is to determine if the DNS resolver already has a translation for the requested name. If the argument at decision block 303 is false, meaning the name is not cached, the DNS resolver contacts the DNS server at step 304 and initiates a request. At step 305 the DNS resolver receives an answer (IP address), a table, an applet, or an error message from the DNS server. Depending on the host name being translated, and the embodiment of the invention, the DNS server could provide any of several possible responses. If the host name is for a site that does not use a distributed web server system, the DNS server returns an answer that is an IP address. Even when the host name is for a site that uses a distributed web server system, the DNS server could return an IP address in one embodiment of the invention, if it is desired to have processing and choices made at the DNS server. In another embodiment of the invention, the DNS server

returns a table or an applet for a host name that is for a web site using a distributed web server system. After step 305, the system proceeds to step 306.

If the argument at decision block 303 is true, meaning that the DNS resolver of the browser does have the name cached, the system proceeds to decision block 306. Note that if the name is cached, it could be translated to either an IP address (answer) or a table/applet. At decision block 306 the argument "Answer or Table/Applet?" is made. If the DNS response (or cached translation) is an answer, the DNS resolver provides the answer (IP address) to the browser at step 307.

If the DNS response is a table/applet the system proceeds to decision block 308. At decision block 308 it is determined whether an applet executable or a table has been returned. If it is a table, the client finds its own client IP address in the provided table at step 309 (this may involve finding a specific entry for its IP address, or a range of IP addresses that contain its own IP address). At step 310 the client links to the web server IP address returned based on its own client IP address.

If at decision block 308 the client determines it has received an applet, the client executes the applet bytecode at step 311 to retrieve a server IP address. The client then links to the retrieved web server IP address at step 312.

Whether using a table or an applet, the client can identify an appropriate server based on a number of factors. For example, the client may be presented with several servers to choose from, with better service available at higher cost (at a fee based web site for example). The client can predetermine criteria for selecting an appropriate server. A client may define connection speed as the most important factor in choosing a server,

irrespective of cost. Alternately, a client could define a minimum connection speed that might be met by servers of different costs at certain times of day. In other cases, the client could preclude servers that have any extra cost associated and therefore only choose from no-cost servers. Clients can also choose a server based on geographical proximity to the client, or on network proximity to the client.

The steps of finding the web server IP address may be accomplished by a simple look up, or by choosing from several possible IP addresses based on the lookup, or after a locality determination as outlined above.

Embodiment of Computer Execution Environment (Hardware)

An embodiment of the invention can be implemented as computer software in the form of computer readable code executed on a general purpose computer such as computer 100 illustrated in Figure 1, or in the form of bytecode class files executable within a Java™ runtime environment running on such a computer. A keyboard 110 and mouse 111 are coupled to a bi-directional system bus 118. The keyboard and mouse are for introducing user input to the computer system and communicating that user input to processor 113. Other suitable input devices may be used in addition to, or in place of, the mouse 111 and keyboard 110. I/O (input/output) unit 119 coupled to bi-directional system bus 118 represents such I/O elements as a printer, A/V (audio/video) I/O, etc.

Computer 100 includes a video memory 114, main memory 115 and mass storage 112, all coupled to bi-directional system bus 118 along with keyboard 110, mouse 111 and processor 113. The mass storage 112 may include both fixed and removable media, such as magnetic, optical or

magnetic optical storage systems or any other available mass storage technology. Bus 118 may contain, for example, thirty-two address lines for addressing video memory 114 or main memory 115. The system bus 118 also includes, for example, a 32-bit data bus for transferring data between and among the components, such as processor 113, main memory 115, video memory 114 and mass storage 112. Alternatively, multiplex data/address lines may be used instead of separate data and address lines.

In one embodiment of the invention, the processor 113 is a microprocessor manufactured by Motorola, such as the 680X0 processor or a microprocessor manufactured by Intel, such as the 80X86, or Pentium processor, or a SPARC™ microprocessor from Sun Microsystems™, Inc. However, any other suitable microprocessor or microcomputer may be utilized. Main memory 115 is comprised of dynamic random access memory (DRAM). Video memory 114 is a dual-ported video random access memory. One port of the video memory 114 is coupled to video amplifier 116. The video amplifier 116 is used to drive the cathode ray tube (CRT) raster monitor 117. Alternatively, the video memory could be used to drive a flat panel or liquid crystal display (LCD), or any other suitable data presentation device. Video amplifier 116 is well known in the art and may be implemented by any suitable apparatus. This circuitry converts pixel data stored in video memory 114 to a raster signal suitable for use by monitor 117. Monitor 117 is a type of monitor suitable for displaying graphic images.

Computer 100 may also include a communication interface 120 coupled to bus 118. Communication interface 120 provides a two-way data communication coupling via a network link 121 to a local network 122. For example, if communication interface 120 is an integrated services digital

network (ISDN) card or a modem, communication interface 120 provides a data communication connection to the corresponding type of telephone line, which comprises part of network link 121. If communication interface 120 is a local area network (LAN) card, communication interface 120 provides a data communication connection via network link 121 to a compatible LAN. Wireless links, modems, or cable modem links are also possible. In any such implementation, communication interface 120 sends and receives electrical, electromagnetic or optical signals which carry digital data streams representing various types of information.

Network link 121 typically provides data communication through one or more networks to other data devices. For example, network link 121 may provide a connection through local network 122 to local server computer 123 or to data equipment operated by an Internet Service Provider (ISP) 124. ISP 124 in turn provides data communication services through the world wide packet data communication network now commonly referred to as the "Internet" 125. Local network 122 and Internet 125 both use electrical, electromagnetic or optical signals which carry digital data streams. The signals through the various networks and the signals on network link 121 and through communication interface 120, which carry the digital data to and from computer 100, are exemplary forms of carrier waves transporting the information.

Computer 100 can send messages and receive data, including program code, through the network(s), network link 121, and communication interface 120. In the Internet example, remote server computer 126 might transmit a requested code for an application program through Internet 125, ISP 124, local network 122 and communication interface 120.

The received code may be executed by processor 113 as it is received, and/or stored in mass storage 112, or other non-volatile storage for later execution. In this manner, computer 100 may obtain application code in the form of a carrier wave.

Application code may be embodied in any form of computer program product. A computer program product comprises a medium configured to store or transport computer readable code, or in which computer readable code may be embedded. Some examples of computer program products are CD-ROM disks, ROM cards, floppy disks, magnetic tapes, computer hard drives, servers on a network, and carrier waves.

The computer systems described above are for purposes of example only. An embodiment of the invention may be implemented in any type of computer system or programming or processing environment.

Thus, a method and apparatus for providing effective traffic localization through the Domain Name System is described.

CLAIMS

1. A method of implementing a distributed server system comprising the steps of:

generating distance/metric information for a web site associated with a distributed server system;

providing said information to a client when said web site is requested by said client;

using said information to identify an appropriate server of said distributed server system to serve said web site to said client.

2. The method of claim 1 wherein said information comprises a plurality of ranges of IP addresses with at least one server IP address associated with each range.

3. The method of claim 1 wherein said step of identifying an appropriate server comprises identifying one of said plurality of ranges of IP addresses containing said client's IP address and selecting said at least one server IP address corresponding to said range.

4. The method of claim 1 wherein said plurality of ranges of IP addresses has a plurality of server IP addresses associated with each range.

5. The method of claim 4 wherein said step of identifying an appropriate server comprises identifying one of said plurality of ranges of IP addresses containing said client's IP address and selecting said one of said plurality of server IP address corresponding to said range.

6. The method of claim 5 wherein said step of selecting one of said plurality of server IP addresses is accomplished by randomly selecting one of said plurality of server IP addresses.

7. The method of claim 5 wherein said step of selecting one of said plurality of server IP addresses is accomplished by identifying a closest one of said plurality of server IP addresses.

8. The method of claim 1 wherein said information comprises a table.

9. The method of claim 1 wherein said information comprises an executable file.

10. The method of claim 9 wherein said executable file comprises an applet.

11. The method of claim 1 wherein said step of generating distance/metric information is accomplished by the use of agents.

12. The method of claim 11 wherein said agents generate said distance/metric information by performing ping triangulations.

13. The method of claim 11 wherein said agents generate said distance/metric information by collecting BGP information.

14. The method of claim 11 wherein said agents generate said distance/metric information by the use of routing tables.

15. The method of claim 5 wherein said step of selecting one of said plurality of server IP addresses is accomplished by determining a cost associated with each of said plurality of server IP addresses and selecting one of said plurality of server IP addresses based said cost.

16. An computer program product having computer readable program code comprising:

computer readable program code configured to cause a computer to generate distance/metric information for a web site associated with a distributed server system;

computer readable program code configured to cause a computer to provide said information to a client when said web site is requested by said client;

computer readable program code configured to cause a computer to use said information to identify an appropriate server of said distributed server system to serve said web site to said client.

17. The computer program product of claim 16 wherein said information comprises a plurality of ranges of IP addresses with at least one server IP address associated with each range.

18. The method of claim 16 wherein identifying an appropriate server comprises identifying one of said plurality of ranges of IP addresses containing said client's IP address and selecting said at least one server IP address corresponding to said range.

19. The computer program product of claim 16 wherein said plurality of ranges of IP addresses has a plurality of server IP addresses associated with each range.

20. The computer program product of claim 19 wherein identifying an appropriate server comprises identifying one of said plurality of ranges of IP addresses containing said client's IP address and selecting said one of said plurality of server IP address corresponding to said range.

21. The computer program product of claim 20 wherein selecting one of said plurality of server IP addresses is accomplished by randomly selecting one of said plurality of server IP addresses.

22. The computer program product of claim 20 wherein selecting one of said plurality of server IP addresses is accomplished by identifying a closest one of said plurality of server IP addresses.

23. The computer program product of claim 16 wherein said information comprises a table.

24. The computer program product of claim 16 wherein said information comprises an executable file.

25. The computer program product of claim 24 wherein said executable file comprises an applet.

26. The computer program product of claim 16 wherein generating distance/metric information is accomplished by the use of agents.

27. The computer program product of claim 26 wherein said agents generate said distance/metric information by performing ping triangulations.

28. The computer program product of claim 26 wherein said agents generate said distance/metric information by collecting BGP information.

29. The computer program product of claim 26 wherein said agents generate said distance/metric information by the use of routing tables.

30. The computer program product of claim 20 wherein selecting one of said plurality of server IP addresses is accomplished by determining a cost associated with each of said plurality of server IP addresses and selecting one of said plurality of server IP addresses based said cost.

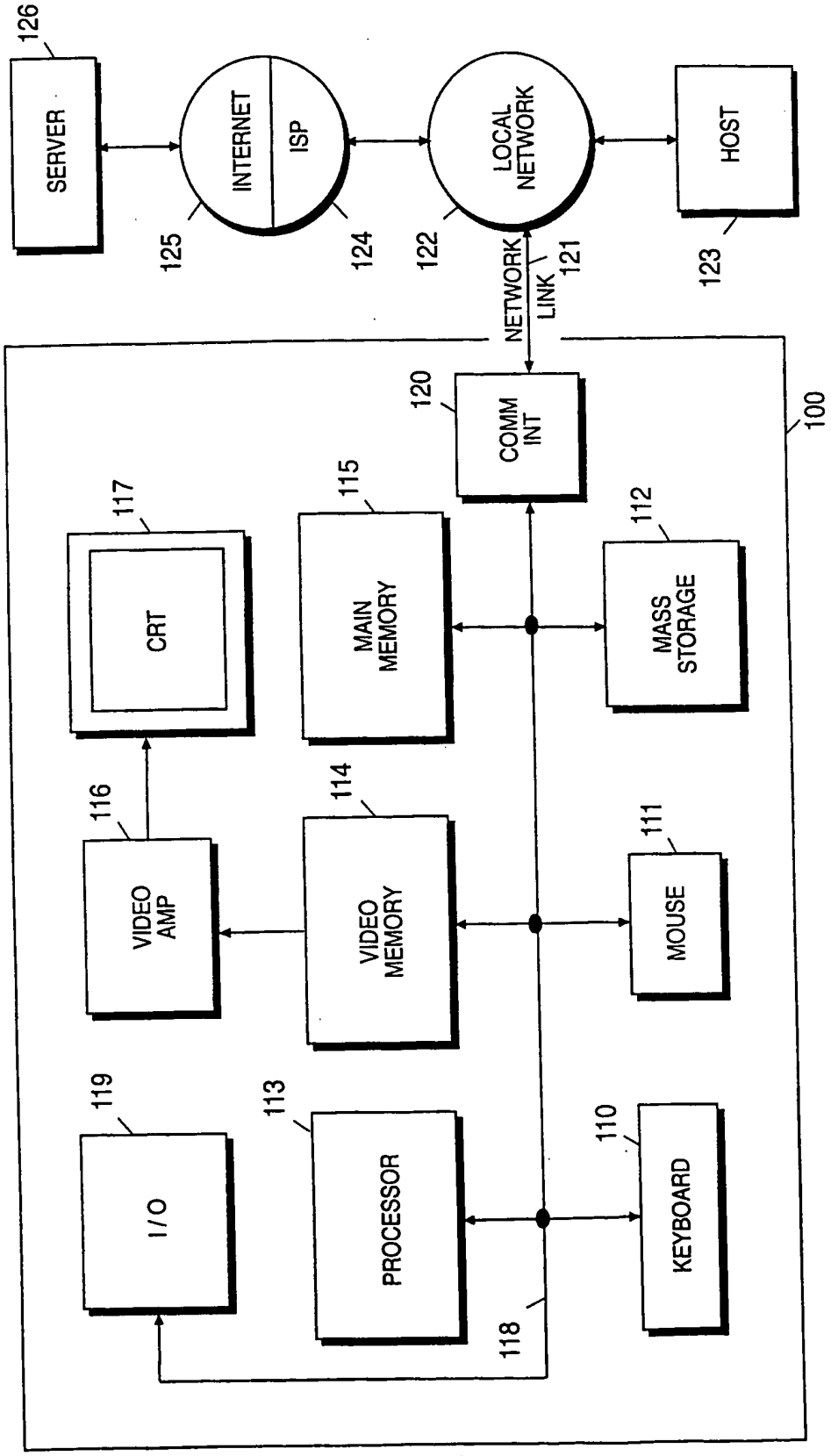
126
SERVER
INTERNET
ISP
125
124
LOCAL NETWORK
122
HOST
123
NETWORK LINK
121
100
120
COMM INT
115
MAIN MEMORY
112
MASS STORAGE
117
CRT
116
VIDEO AMP
114
VIDEO MEMORY
111
MOUSE
119
I/O
113
PROCESSOR
110
KEYBOARD
118

FIG. 1

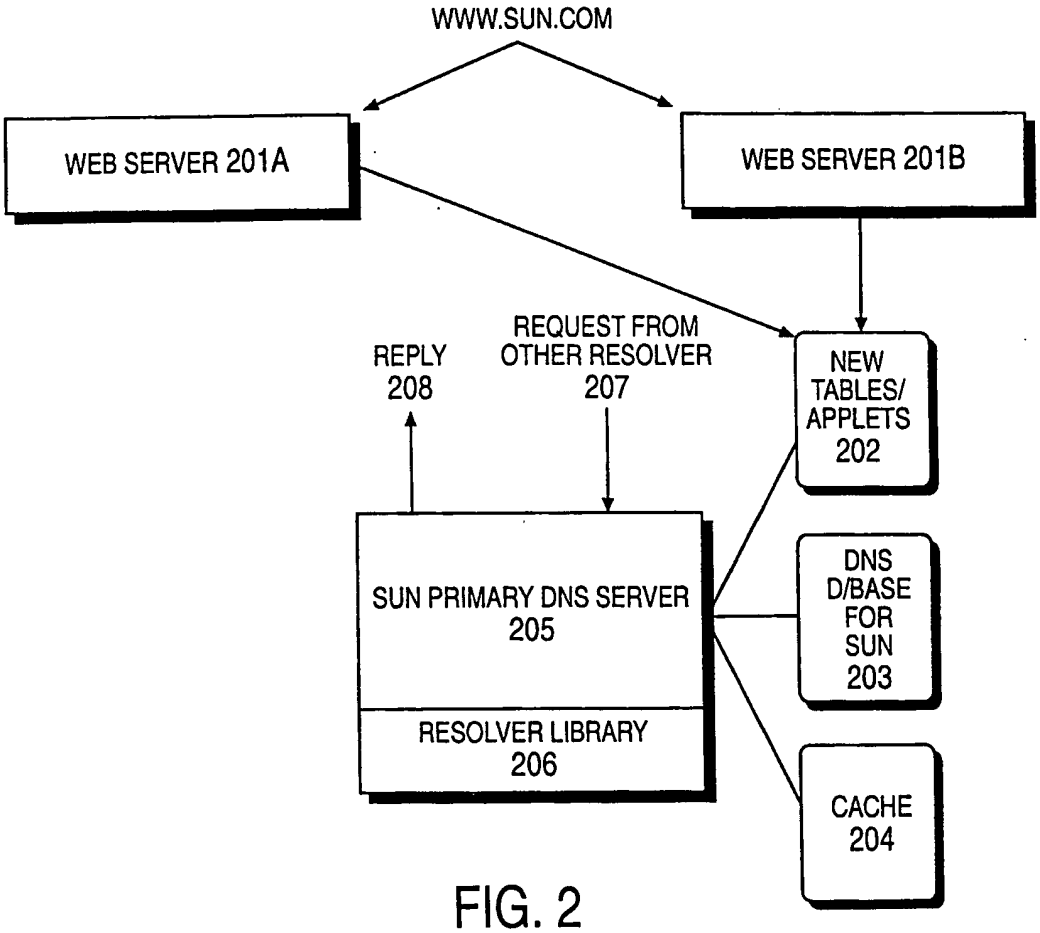
WWW.SUN.COM
WEB SERVER 201A
WEB SERVER 201B
REPLY
208
REQUEST FROM
OTHER RESOLVER
207
NEW
TABLES/
APPLETS
202
SUN PRIMARY DNS SERVER
205
RESOLVER LIBRARY
206
DNS
D/BASE
FOR
SUN
203
CACHE
204

FIG. 2

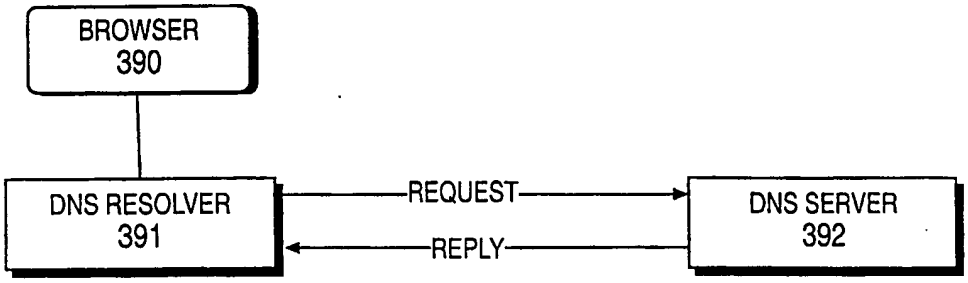

FIG. 3A

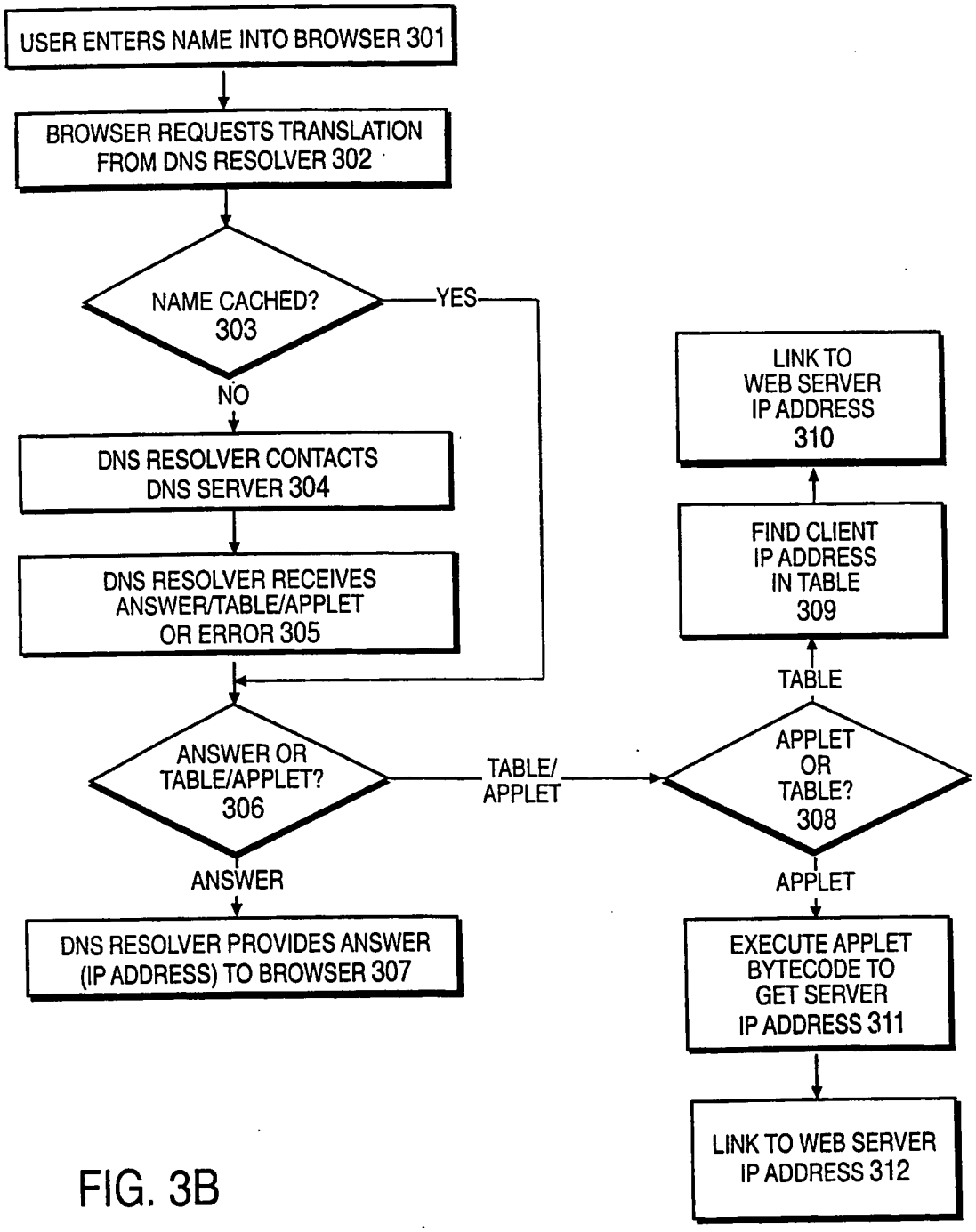
USER ENTERS NAME INTO BROWSER 301
BROWSER REQUESTS TRANSLATION FROM DNS RESOLVER 302
NAME CACHED? 303
YES
NO
DNS RESOLVER CONTACTS DNS SERVER 304
DNS RESOLVER RECEIVES ANSWER/TABLE/APPLET OR ERROR 305
ANSWER OR TABLE/APPLET? 306
TABLE/ APPLET
ANSWER
DNS RESOLVER PROVIDES ANSWER (IP ADDRESS) TO BROWSER 307
APPLET OR TABLE? 308
TABLE
APPLET
FIND CLIENT IP ADDRESS IN TABLE 309
LINK TO WEB SERVER IP ADDRESS 310
EXECUTE APPLET BYTECODE TO GET SERVER IP ADDRESS 311
LINK TO WEB SERVER IP ADDRESS 312

FIG. 3B